An Adaptive Scheduling Algorithm for TDM Switching Systems

Wen-Tsuen Chen, *Fellow, IEEE*, and Huai-Jen Liu

Abstract— **In this paper, we consider the scheduling problem in time-division multiplexed (TDM) switching systems. In previous works, the interdependence between traffic demands in two consecutive frames is neglected, and scheduling algorithms found up to now have time complexities $O(N^5)$ or $O(N^{4.5})$, where N is the switch size. However, in many applications like voice or video communications, if a source transmits a packet to a destination in a frame, it is highly probable that it will also transmit a packet to the same destination in the next frame. So it is not necessary to schedule incoming packets for every frame if we can preserve all the switching patterns for the nearest scheduled frame and update the patterns appropriately according to the changes of traffic demands. The adaptive algorithm proposed in this paper assigns time slots to packets according to the changes of traffic demands. This algorithm has the worst case time complexity $O(N^2L)$, where L is the TDM frame length. Comparing the time complexity of the adaptive algorithm with those of previous scheduling algorithms, the adaptive algorithm can perform better than previous scheduling algorithms when N is large and/or L is small. Since traffic demands in consecutive frames are expected to be interdependent in many applications, the proposed algorithm may offer as an efficient alternative for scheduling time slots in these applications.**

I. Introduction

Time-division multiplexed (TDM) switching has widely been employed in satellite communications [23] and terrestrial networks [14]. In terrestrial networks, a time-division packet switching system may have a switch in which the internal switch elements are set to establish paths from input lines to output lines. In a satellite-switched time-division multiple access (SS/TDMA) system, the satellite has several spot-beam antennas and a solid-state RF switch. The antennas are directed toward spatially disjoint geographical zones and the switch provides connections from up-link beams to down-link beams through on-board transponders. Traffic from an earth station in the beam zone covered by an up-link beam is routed by a transponder to an earth station in the beam zone covered by a down-link beam.

Paper approved by Inder Gopal, the Editor for Network Protocols of the IEEE Communications Society. Manuscript received: October 25, 1990; revised February 7, 1993. This work was supported by the National Science Council, Taiwan, Republic of China, under grant NSC79-0408-E-007-16. This paper was presented in part at the IEEE INFOCOM'91 Conference, Miami, Florida, April 1991.

W. T. Chen is with the Department of Computer Science, National Tsing Hua University, Hsin-Chu, Taiwan 30043, Republic of China.

H. J. Liu is with the Switch Technology Laboratory, Telecommunication Laboratories, Yang-Mei, Taiwan 32617, Republic of China.

IEEE Log Number 9410098.

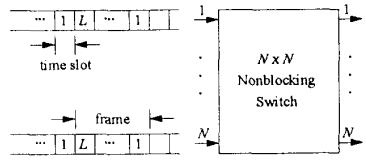

Fig. 1. An $N \times N$ switching system.

In both cases, the TDM switching system can be represented as shown in Fig. 1. There are N input lines and N output lines, where N is referred to as the *switch size*. Traffic in a TDM line is assumed to be allocated in repetitive *frames* each of which has L *time slots*, where L is referred to as the TDM *frame length* or the *capacity* of a TDM line. Time slots are synchronized and shared by users. Sources at the inputs can communicate with destinations at the outputs by asking for different amount of time slots to transmit packets to destinations. Each new traffic demand can be accepted if the input line and the output line both have enough free time slots. There are two kinds of possible traffic conflicts in a TDM switching system; namely *internal conflicts* and *output conflicts*. If more than one packet are routed through the same internal link of the switch in the same time slot, an internal conflict occurs. If more than one packet are destined for the same output line in the same time slot, an output conflict occurs. Both kinds of traffic conflicts may result in loss of packets. In this paper, we assume that the switch is a nonblocking switch and therefore there are no internal conflicts [7]. We shall consider the scheduling problem to avoid output conflicts. Generally connection requests arise without coordination, so it is necessary to ensure that incoming packets from different TDM input lines should not be destined for the same output line in the same time slot. The task to scheduling packets is performed by the network controller to avoid conflicts at the outputs. Therefore each source conducts its TDM transmission according to the time slot assignment announced by the network controller.

The scheduling problem to avoid output conflicts can be solved by using the time slot assignment algorithms developed in [1]-[4], [8], [13], [17], [19]-[21]. These time slot assignment algorithms are essentially to find a sequence of switching configurations to transmit packets in each frame with the objective that the overall transmission duration is minimum. A switching configuration is a set of switch settings by which packets can be transmitted without conflicts. These algorithms find a switching configuration by

calling some combinatorial optimization algorithms, such as algorithms for finding a system of distinct representative (SDR) [13], [20], [21], finding a max-min bipartite matching [2], [3], [8], finding a maximum cardinality matching [17], and finding a maximum weight matching [19] etc. The number of switching configurations needed in each frame is at most $O(N^2)$ [2]-[4], [13], [19], [20], and these combinatorial algorithms are of time complexities $O(N^3)$ [16] or $O(N^{2.5})$ [11]. So the time complexities of these time slot assignment algorithms are $O(N^5)$ or $O(N^{4.5})$. Although they are polynomial time algorithms, their time complexities are still so high that they are not very suitable in many practical applications. The computational bottleneck may be overcome by using special-purpose parallel processors to reduce the time spent in finding a switching configuration [21]. However, the number of processors increases with $O(N^2)$. This may incur extra cost of the switch in a TDM system. In this paper, we shall propose a scheduling algorithm with a lower time complexity.

In previous works [1]-[4], [6], [8]-[10], [13], [17], [19]-[21], the interdependence between traffic demands in two consecutive frames is neglected. However, in many practical applications, if a source transmits a packet to a destination in a frame, it is highly probable that it will also transmit a packet to the same destination in the next frame unless the connection from the source to the destination is no longer needed. For example, consider the video communications which have the following characteristics:

- Packets in video communications are generated at their sources at a constant rate.
- Because of the real-time requirement, a video packet is considered to have a higher priority than a data packet has. In general, packets of lower priorities are taken as space fillers in the transmission of packets of higher priorities.
- Each video connection usually continues for a number of consecutive frames.

With these characteristics, traffic demands between two consecutive frames is correlated. So it is not necessary to schedule incoming packets for every frame if we can preserve all the switching patterns for the nearest scheduled frame and update the patterns appropriately according to the changes of traffic demands.

In this paper, we shall give an adaptive algorithm which assigns time slots to packets according to the changes of traffic demands. The amount of changes of traffic demands will therefore affect the performance of the adaptive algorithm. It is shown that the adaptive scheduling algorithm will be faster than previous time slot assignment algorithms with interdependent traffic since the amount of changes of traffic demands between any two consecutive frames may be small.

In the following, we shall present the mathematical formulation of the scheduling problem in Section II. The adaptive scheduling algorithm will be proposed in Section III. A probability model with interdependent traffic is used to analyze the amount of changes of traffic demands in Section IV. Also included in Section IV are computer simulation results with interdependent traffic. Some conclusions are made in Section V.

II. Problem Formulation

The system under consideration is shown in Fig. 1 with an $N \times N$ nonblocking switch. Each input line is allowed L time slots per frame to transmit packets to destinations at the outputs, and each output line is also allowed L time slots per frame to receive packets from sources at the inputs. The traffic demands in a frame can be characterized by an $N \times N$ *traffic matrix* T. An entry $t_{i,j}$ in T represents the traffic demands from input i to output j in the frame, measured in number of packets. We assume that exactly one packet can be transmitted in a time slot. Therefore $t_{i,j}$ also represents the number of time slots needed to transmit packets from input i to output j in the frame. A switching configuration in the r-th time slot can be characterized by an $N \times N$ *switching matrix* S_r. An entry $s_{r_{i,j}} = 1$ in S_r represents that the source at input i can transmit a packet in the r-th time slot to the destination at output j if the switching pattern is set according to S_r in the r-th time slot. Hence a packet from input i to output j, denoted as $P_{i,j}$, is said to be allocated in the r-th time slot, or to be allocated in S_r. The r-th time slot is said to be assigned to or reserved for packets corresponding to those unities in S_r. Since it is impossible for a source to transmit more than one packet in a time slot, there is no more than one unity in a row of a switching matrix. In addition, there is no more than one unity in a column of a switching matrix; otherwise, output conflicts may occur at the output corresponding to this column. So it is necessary for a switching matrix S_r to have exactly one unity in each row and each column. All the other entries are 0's.

Given an $N \times N$ traffic matrix T, we call a row or a column a *line*. The sum of entries in a line is called the *line sum*. The line sum of a row i (column j), denoted by R_i (C_j), is called a *row (column) sum*, where $1 \le i \le N$ ($1 \le j \le N$). Since a new traffic demand from input i to output j can be accepted if the frames at input i and output j have enough free time slots,

1. the number of packets transmitted by a given input i can not exceed L in a given frame, i.e., $R_i \le L$, and
2. the number of packets destined for a particular output j can not exceed L in a given frame, i.e., $C_j \le L$.

The above conditions are called the *scheduling criteria* [1], [21], [22]. It has been shown that if T satisfies the above scheduling criteria then it is possible to scheduling incoming packets such that no conflicts occur [1]. The scheduling problem is then defined as follows: Given a traffic matrix T which satisfies the scheduling criteria, find a sequence of L switching matrices $\{S_r\}$ such that each packet of the traffic matrix T can be allocated in an S_r, i.e., $\sum_{r=1}^{L} s_{r_{i,j}} \ge t_{i,j}$ for all i and j, $1 \le i, j \le N$. The sequence $\{S_r\}$ is called a *schedule* of T.

As mentioned earlier, in many practical applications, traffic demands between two consecutive frames are interdependent. Most of traffic demands in a given frame may be the same as those in the frame coming before it. If we can preserve all the switching patterns for the nearest scheduled frame and update the switching patterns appropriately according to the changes of traffic demands, it is expected that the scheduling time can be reduced. Based on this concept, we define the adaptive scheduling problem as follows: Given a traffic matrix T which satisfies the scheduling criteria and a sequence of L switching matrices $\{S_r'\}$, find switching matrices $\{S_r\}$ by updating $\{S_r'\}$ such that the sequence $\{S_r\}$ is a scheduling of T.

An $N \times N$ traffic matrix T is said to be *complete with respect to* L or *complete* for short, if all the line sums of T are equal to L. That is, T is complete if $R_1 = ... = R_N = C_1 = ... = C_N = L$. Note that a complete traffic matrix satisfies the scheduling criteria. Also we can not add any new traffic demand to a complete traffic matrix; otherwise, the scheduling criteria is violated. It has been shown that the sequence $\{S_r\}$ is a schedule of a complete traffic matrix T if and only if $T = \sum_{r=1}^{L} S_r$, i.e., $\sum_{r=1}^{L} s_{r_{i,j}} = t_{i,j}$ for all i and j, $1 \le i, j \le N$ [1], [2], [4], [13]. The sequence $\{S_r\}$ is called an *optimal schedule* of the complete traffic matrix T.

It has been shown that, given any traffic matrix T which satisfies the scheduling criteria, it is always possible to add appropriate non-negative integers (called *dummy traffic*) to the entries in T such that the resulting traffic matrix, denoted as T^c, is complete with respect to L [1], [2], [4], [13]. An optimal schedule of T^c is obviously a schedule of T [1], [2], [4], [13]. So the adaptive scheduling problem with a traffic matrix T can be solved if the adaptive scheduling problem with the corresponding traffic matrix T^c is solved. In this paper, we shall follow this approach to find a schedule of T by finding an optimal schedule of T^c. The assignments corresponding to the dummy traffic can be ignored, and simply reflect the fact that the available capacity of the switching system exceeds the total accepted traffic demands.

III. Algorithm

In this section, we shall first describe the basic idea behind the adaptive scheduling (AS) algorithm by showing how to adaptively assign time slots to packets according to the changes of traffic demands. The algorithm is then given, together with the analysis of its time complexity.

A. Basic Idea

We first introduce some notations to be used in the AS algorithm. If an entry $x_{i,j}$ of a matrix is equal to 1, 0, or -1, we denote it by $1_{i,j}$, $0_{i,j}$, or $-1_{i,j}$, respectively. If there is only one non-zero entry $1_{i,j}$ or $-1_{i,j}$ in a matrix, we denote this matrix by $[1]_{i,j}$ or $[-1]_{i,j}$, respectively.

Given a schedule $\{S_r'\}$ for a frame and a traffic matrix T which satisfies the scheduling criteria, we want to find a schedule $\{S_r\}$ of T by updating $\{S_r'\}$. Let T' be $\sum_{r=1}^{L} S_r'$. Then T' is a complete traffic matrix corresponding to the scheduled frame and $\{S_r'\}$ is an optimal schedule of T'.

First, we add dummy traffic to T such that the resulting traffic matrix T^c is complete with respect to L. Then we initiate the matrix V with $T^c - T'$. If we can update the switching matrices $\{S_r'\}$ such that V becomes a zero matrix, the resulting sequence of switching matrices $\{S_r\}$ is an optimal schedule of T^c and hence a schedule of T.

We observe that each line sum of V is zero. This is because each line sum of T^c and T' is equal to L. If there exists a negative entry $v_{i,j}$ for some i and j, then there exists a positive entry $v_{i,q}$ for some q since the i-the row sum of V is zero. Similarly there is also a negative entry $v_{p,q}$ for some p since the q-th column sum of V is zero. The positive entry $v_{i,q}$ indicates that there are $v_{i,q}$ new traffic demands, measured in number of packets, from input i to output q needed to be allocated in the current frame. The negative entry $v_{i,j}$ indicates that there are $|v_{i,j}|$, the absolute value of $v_{i,j}$, time slots which are assigned in the previously scheduled frame to traffic demands from input i to output j and no longer needed in the current frame. So they can be deallocated. Similarly, the negative entry $v_{p,q}$ indicates that there are $|v_{p,q}|$ time slots which are assigned in the previously scheduled frame to traffic demands from input p to output q and no longer needed in the current frame. Also they can be deallocated. Hence there exists a switching matrix S_x' containing an entry $1_{i,j}$, and there exists a switching matrix S_y' containing an entry $1_{p,q}$. Then, by calling the reallocation algorithm described in the following, we shall deallocate $\boldsymbol{P}_{i,j}$ from the x-th time slot and $\boldsymbol{P}_{p,q}$ from the y-th time slot, and allocate $\boldsymbol{P}_{i,q}$ in the y-th time slot. Then S_x' and S_y' are subsequently updated such that (1) the resulting matrices S_x and S_y are still switching matrices, and (2) $S_x + S_y - S_x' - S_y' = [-1]_{i,j} + [-1]_{p,q} + [1]_{i,q} + [1]_{p,j}$. The last expression means that we let time slots originally assigned to $\boldsymbol{P}_{i,j}$ and $\boldsymbol{P}_{p,q}$ be reserved for $\boldsymbol{P}_{i,q}$ and $\boldsymbol{P}_{p,j}$. If there is no newly accepted traffic demand from input p to output j, the time slot just reserved for $\boldsymbol{P}_{p,j}$ will make $v_{p,j}$ negative. After updating these two switching matrices, V is updated such that each line sum of V is still kept zero. The above procedure is repeated until V is full of zero entries. Then the resulting sequence of the switching matrices is an optimal schedule of T^c and hence a schedule of T. In the following, we shall first describe the reallocation algorithm.

B. Reallocation Algorithm

The reallocation algorithm essentially first deallocates $\boldsymbol{P}_{i,j}$ from S_x and $\boldsymbol{P}_{p,q}$ from S_y, and allocates $\boldsymbol{P}_{i,q}$ in S_y. Then the packet in conflict with $\boldsymbol{P}_{i,q}$ is reallocated from S_y to S_x. This reallocation may further introduce a conflict in S_x. Then the packet in conflict with the reallocated packet is reallocated from S_x to S_y. This procedure is repeated until no conflict exists. $\boldsymbol{P}_{p,j}$ is finally allocated. The reallocation algorithm is given as follows:

Reallocation Algorithm

Input : $i, j, p, q, i \neq p$ and $j \neq q$, and two switching matrices S_x' and S_y', $x \neq y$, where S_x' contains an entry $1_{i,j}$ and S_y' contains an entry $1_{p,q}$.

Output: Two switching matrices S_x and S_y resulting from updating S_x' and S_y', respectively, such that $S_x + S_y - S_x' - S_y' = [-1]_{i,j} + [-1]_{p,q} + [1]_{i,q} + [1]_{p,j}$.

Procedure:

Step 1. $S_x \leftarrow S_x'$.
$S_y \leftarrow S_y'$.

Step 2. $s_{x_{i,j}} \leftarrow 0$. /* Deallocate $\boldsymbol{P}_{i,j}$ from S_x. */
$s_{y_{p,q}} \leftarrow 0$. /* Deallocate $\boldsymbol{P}_{p,q}$ from S_y. */
$s_{y_{i,q}} \leftarrow 1$. /* Allocate $\boldsymbol{P}_{i,q}$ in S_y. */

Step 3. $g \leftarrow i$.
$h \leftarrow q$.

Step 4. /* Find $\boldsymbol{P}_{g,k}$ in row g of S_y that is conflict with $\boldsymbol{P}_{g,h}$, and reallocate it from S_y to S_x. */
Find column k such that $s_{y_{g,k}} = 1$ and $k \neq h$.
$s_{y_{g,k}} \leftarrow 0$.
$s_{x_{g,k}} \leftarrow 1$.

Step 5. /* If $k = j$, then allocate $\boldsymbol{P}_{p,j}$ in S_y. */
If $k = j$, then $s_{y_{p,j}} \leftarrow 1$ and stop.

Step 6. /* Find $\boldsymbol{P}_{m,k}$ in column k of S_x that is in conflict with $\boldsymbol{P}_{g,k}$, and reallocate it from S_x to S_y. */
Find row m such that $s_{x_{m,k}} = 1$ and $m \neq g$.
$s_{x_{m,k}} \leftarrow 0$.
$s_{y_{m,k}} \leftarrow 1$.

Step 7. /* If $m = p$, then allocate $\boldsymbol{P}_{p,j}$ in S_x. */
If $m = p$, then $s_{x_{p,j}} \leftarrow 1$ and stop.

Step 8. $g \leftarrow m$.
$h \leftarrow k$.
Go To Step 4.

We have proved the following theorem in [5].

Theorem 1: After applying the reallocation algorithm, the matrices S_x and S_y resulting from updating S_x' and S_y', respectively, are still switching matrices, and $S_x + S_y - S_x' - S_y' = [-1]_{i,j} + [-1]_{p,q} + [1]_{i,q} + [1]_{p,j}$.

Thus, the output of the reallocation algorithm is correct.

C. Adaptive Scheduling Algorithm

We now give the detailed adaptive scheduling algorithm in the following.

Adaptive Scheduling (AS) Algorithm

Input : (1) A traffic matrix T, which satisfies the scheduling criteria in a frame of length L, (2) a scheduled traffic matrix T' which is complete with respect to L, and (3) an optimal schedule $\{S_r'\}$ of T', where T' and T are traffic matrices of two consecutive frames and the frame corresponding to T' comes before the frame corresponding to T.

Output: An optimal schedule $\{S_r\}$ of T.

Procedure:

Step 1. By calling the filling algorithm [4], add dummy traffic to T such that T is complete with respect to L.

Step 2. V is initiated with $T - T'$, and $\{S_r\}$ is initiated with $\{S_r'\}$. The entry r_i^- of the column vector R^- is initiated with the sum of negative entries in row i of V for all i, $1 \leq i \leq N$.
/* The column vector R^- is used to reduce the time spent in finding negative entries of V from Step 4 to Step 6. The absolute value of r_i^-, $|r_i^-|$, is equal to the number of traffic demands from input i which can be deallocated. Since we shall always keep each line sum of V zero, $|r_i^-|$ also represents the number of traffic demands from input i which are not yet allocated. */

Step 3. $i \leftarrow 1$.

Step 4. /* At the following three steps, find a traffic demand which is no longer needed in the current frame if it exists. */
While $(r_i^- = 0)$ and $(i \leq N)$ do
$i \leftarrow i + 1$
Endwhile

Step 5. /* If $r_i^- = 0$ for all i, $1 \leq i \leq N$, i.e., if there is no negative entry in V, then stop and thus $\{S_r\}$ is an optimal schedule of T. */
If $i > N$ then stop.

Step 6. Find the first column j such that $v_{i,j} < 0$.

Step 7. /* Find a new traffic demand in the current frame from the same input line as the traffic demand found at Step 4. */
Find the first column q such that $v_{i,q} > 0$.

Step 8. /* Find a traffic demand which is destined for the same output line as the traffic demand found at Step 7 and is no longer needed in the current frame. */
Find the first row p such that $v_{p,q} < 0$.

Step 9. /* Find two time slots containing the traffic demands found at Step 6 and Step 8. */
Find a switching matrix S_x such that it contains an entry $1_{i,j}$.
Find a switching matrix S_y such that it contains an entry $1_{p,q}$.

Step 10. If $x = y$
Begin
$s_{x_{i,j}} \leftarrow 0$. /* Deallocate $\boldsymbol{P}_{i,j}$ in S_x. */
$s_{x_{i,q}} \leftarrow 1$. /* Allocate $\boldsymbol{P}_{i,q}$ in S_x. */
$s_{x_{p,q}} \leftarrow 0$. /* Deallocate $\boldsymbol{P}_{p,q}$ in S_x. */
$s_{x_{p,j}} \leftarrow 1$. /* Allocate $\boldsymbol{P}_{p,j}$ in S_x. */
End
Else
Update S_x and S_y by calling the reallocation algorithm with the parameters i, j, p, q, S_x, and S_y.

Step 11. /* Update R^- and V. */

$r_i^- \leftarrow r_i^- + 1$.

/* Note that $P_{p,q}$ in S_x is deallocated at Step 10. Hence the number of traffic demands from input p which can be deallocated should be decreased by 1. That is, r_p^- should be increased by 1. But note that $P_{p,j}$ is also allocated at Step 10. If $v_{p,j} \le 0$, i.e., there is no new traffic demand from input p to output j, then the just allocated $P_{p,j}$ shall be deallocated and hence r_p^- should be decreased by 1. So, if $v_{p,j} \le 0$, r_p^- is unchanged. */

If $v_{p,j} > 0$ then $r_p^- \leftarrow r_p^- + 1$.

$V \leftarrow V + [1]_{i,j} + [1]_{p,q} - [1]_{i,q} - [1]_{p,j}$.

Go To Step 4.

The following theorems have been proved in [5].

Theorem 2: Updating of V at Step 11 is correct, and each line sum of updated V is kept zero.

Theorem 3: V is full of zeros when the AS algorithm terminates.

Hence we know that the AS algorithm will update the switching matrices such that the sequence of these updated switching matrices is a schedule of the given traffic matrix.

D. Time Complexity

The data structures used to maintain the switching matrices may affect the time complexities of the AS algorithm. We shall analyze it together with those data structures used to reduce the time spent in searching for S_x and S_y at Step 9 in the AS algorithm and in searching for k at Step 4 and m at Step 6 in the reallocation algorithm. Note that the data structures described in the following are only initiated when the switching system is set up. Therefore we do no take into account of the time to set up the data structures when we analyze the time complexity of the AS algorithm.

Consider the reallocation algorithm first. Since a switching matrix S_r is obviously a permutation matrix [18], we can use a two-dimensional array SM to represent the sequence of L switching matrices $\{S_r\}$. $SM(x,i) = j$ if $s_{x_{i,j}} = 1$. So searching for k at Step 4 can be done by the following statement: $k \leftarrow SM(y,g)$, in constant time. In addition, another two-dimensional array ISM is also used to represent the same sequence of L switching matrices $\{S_r\}$. $ISM(x,j) = i$ if $s_{x_{i,j}} = 1$. So searching for m at Step 6 can be done by the following statement: $m \leftarrow ISM(x,k)$, in constant time. The detailed implementation for the reallocation algorithm by using these data structures is not presented in this paper, but it is easy to see that the reallocation of a packet requires updating of the entries of SM and ISM in constant time. Consequently the time complexity of the reallocation algorithm is $O(N)$ since there are at most $2N$ reallocations. Simulation results given in the next section will show that the number of reallocations within the reallocation algorithm is far less than $2N$.

Now consider the AS algorithm. In order to search for S_x and S_y at Step 9 in the AS algorithm in constant time, data structures of doubly linked lists [12] are used. For any unity entry, there are two linking fields, PRE and $SUCC$, associated with it. The PRE field of a unity entry, say $s_{x_{a,b}} = 1$, contains the index of the preceding switching matrix that contains $1_{a,b}$. The $SUCC$ field of $s_{x_{a,b}}$ contains the index of the succeeding switching matrix that contains $1_{a,b}$. The headers of these linked lists are represented by a two-dimensional array LL_FIRST of size $N \times N$. All the switching matrices containing an entry $1_{a,b}$ are linked together by the doubly linked list with the header $LL\_FIRST(a,b)$, for all a and b, $1 \le a, b \le N$. When we search for a switching matrix S_x (S_y) that contains an entry $1_{i,j}$ ($1_{p,q}$) at Step 9, the first switching matrix pointed to by $LL\_FIRST(i,j)$ ($LL\_FIRST(p,q)$) is chosen and is easily found in constant time. The updating of these linked lists is performed when packets are reallocated at Step 10. When a packet $P_{a,b}$ is deallocated from its corresponding switching matrix, say S_x, S_x needs to be deleted from the linked list with the header $LL\_FIRST(a,b)$. Since it is a doubly linked list, it is easy to delete S_x in constant time by updating $SUCC(PRE(x,a),a)$ and $PRE(SUCC(x,a),a)$ [12]. When a packet $P_{a,b}$ is allocated in S_x, S_x needs to be inserted into the linked list with the header $LL\_FIRST(a,b)$. It is simply inserted to the head of this linked list by updating $PRE(LL\_FIRST(a,b),a)$, $LL\_FIRST(a,b)$, and $SUCC(x,a)$ in constant time [12]. Consequently the updating of these linked lists can be done in constant time.

After describing the data structures, we consider the time complexity of each step of the AS algorithm. The preprocessing at Step 1 and Step 2 has time complexity $O(N^2)$. The tasks to search for i, j, p, and q from Step 4 to Step 8 have time complexity $O(N)$. Searching for S_x and S_y at Step 9 can be done by searching linked lists in constant time. The time complexity of Step 10 is dominated by that of the reallocation algorithm and hence is $O(N)$. Finally, consider the number of iterations of the procedure. At each iteration, the sum of the positive entries in the matrix V will be decreased by at least 1 (and the sum of the negative entries in the matrix V will be incremented by at least 1), so the number of iterations is at most equal to the sum of the positive entries, denoted as δ, in V ($= T - T'$) at Step 2. As a result, the AS algorithm has the time complexity $O(\max(N^2, \delta N))$.

In the worst case, all the traffic demands from a given input line are changed such that the sum of the positive entries in each row of V at Step 2 is the frame length L. Hence the maximum value of δ is NL. Consequently, the AS algorithm has the worst case time complexity $O(N^2 L)$.

Comparing the AS algorithm of time complexity $O(N^2 L)$ with previous time slot assignment algorithms of time complexity $O(N^{4.5})$, the algorithm may perform worse than previous time slot assignment algorithms if $L > cN^{2.5}$, where c is a parameter depending on the hardware (the network controller, the switch architecture, the machine wherein

algorithms are implemented, etc.) and the software (the implementations of algorithms, etc.). In other words, the AS algorithm will perform better than previous time slot assignment algorithms if $L < cN^{2.5}$. Consequently the AS algorithm can perform better than previous time slot assignment algorithms when N is large and/or L is small in the worst case.

Since, in many applications, most of connections will continue for a number of consecutive frames, the amount of changes of traffic demands δ will be far less than NL and hence the number of total reallocations will be far less than $2N^2L$, the maximum number of total reallocations. So the AS algorithm is expected to be faster than previous time slot assignment algorithms in many practical applications.

IV. Probability Model and Simulation Results

The time complexity of the AS algorithm is dominated by the number of total reallocations which depends on the amount of changes of traffic demands δ and the number of reallocations within the reallocation algorithm. A probability model with interdependent traffic is used to analyze the amount of changes of traffic demands δ. This model is validated by computer simulations. The number of reallocations within the reallocation algorithm is obtained through simulations.

A. Probability Model

The connection requests from an input line are assumed to be generated according to a Poisson distribution with mean λ, measured in number of connection requests per time slot, and the connection holding time is assumed to be exponentially distributed with mean $1/\mu$, measured in number of time slots per connection. The probability that a connection request is destined for a given output line is $1/N$. Hence the connection requests from input i to output j are generated according to a Poisson distribution with mean λ/N. For convenience, let

$$\lambda^c = \lambda / N. \tag{1}$$

Since we assume the probability that a connection request is destined for any given output is identical, the expected capacity from a given input to a given output is L/N time slots. So, as the ratio L/N becomes large, correlation caused by enforcement of the scheduling criteria among inputs and outputs becomes negligible. This implies that, as L/N becomes large, $t_{i,j}$ can be viewed as an independent probability random variable, where $t_{i,j}$ is the number of traffic demands from input i to output j. For small L/N, the correlation among inputs and outputs can not be neglected. Due to this correlation, the number of unusable time slots will increase [22]. Hence the number of accepted connection requests will be less than the expected number of traffic demands when L/N is small. This implies that the excepted amount of changes of traffic demands $E[\delta]$, obtained based on the assumption of independence among inputs and outputs, can give an upper bound for actual δ on the average.

Suppose T and T' are two consecutive traffic matrices and the frame corresponding to T' precedes the frame corresponding to T. Let $V = T - T'$ and let

$$v^+_{i,j} = \begin{cases} v_{i,j} = t_{i,j} - t'_{i,j} & \text{if } t_{i,j} \ge t'_{i,j} \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Since the time taken by the AS algorithm depends on the sum of the positive entries in V, we shall find the mean value of the sum of the positive entries in V, $E[\delta]$, which is given as follows:

$$E[\delta] = N^2 E[v^+_{i,j}]$$
$$= N^2 \sum_{n=0}^{L} \Pr[t'_{i,j} = n] \sum_{k=0}^{L-n} k \Pr[t_{i,j} = n + k | t'_{i,j} = n], \tag{3}$$

as L/N becomes large.

Assume that the switching system is stationary. So the limiting probabilities $\Pr[t_{i,j}' = n] = \Pr[t_{i,j} = n]$ for all n, i, and j, $0 \le n \le L$ and $1 \le i, j \le N$. For convenience, let $\mathbf{Q}$ denote the transition probability matrix with the elements $q_{m,n} = \Pr[t_{i,j} = n | t_{i,j}' = m]$ and let Π denote the probability vector with the elements $\pi_m = \Pr[t_{i,j} = m]$ for all m and n, $0 \le m, n \le L$. So we have the following equations [15]

$$\Pi = \Pi \mathbf{Q} \tag{4}$$

and

$$\sum_{m=0}^{L} \pi_m = 1. \tag{5}$$

Hence

$$E[\delta] = N^2 \sum_{n=0}^{L} \pi_n \sum_{k=0}^{L-n} k q_{n,n+k}. \tag{6}$$

In the following, we shall derive the transition probabilities $q_{n,n+k}$ and hence the limiting probabilities π_k can be obtained from (4) and (5). Finally $E[\delta]$ is obtained from (6).

The probability that, given $t_{i,j}' = n$, m connections will be disconnected in the frame corresponding to T is

$$\binom{n}{m} (1 - e^{-\mu L})^m (e^{-\mu L})^{n-m}, \quad 0 \le m \le n. \tag{7}$$

The probability that, as L/N becomes large, there are $m+k$ connection requests from input i to output j generated in the frame corresponding to T is

$$\frac{(\lambda^c L)^{m+k} e^{-\lambda^c L}}{(m+k)!}, \quad m + k \ge 0. \tag{8}$$

Hence, $q_{n,n+k}$ is computed from (9) as follows:

$$q_{n,n+k} = \begin{cases} \sum_{m=\max(-k,0)}^{n} \binom{n}{m} (1 - e^{-\mu L})^m (e^{-\mu L})^{n-m} \dfrac{(\lambda^c L)^{m+k} e^{-\lambda^c L}}{(m+k)!} & \text{if } 0 \le n + k < L \\ 1 - \sum_{m=0}^{L-1} q_{n,m} & \text{if } n + k = L. \end{cases} \tag{9}$$

Simulations are conducted to test the validity of the probability model with various values of L and N. Given N = 20, $\lambda = 0.1$, and $\mu = 0.1$, the analytic and simulation results are the same with $L \geq 200$, as shown in Fig. 2. With $L < 200$, the analytic and simulation results are not the same but they are very close such that their values are almost equal in logarithmic scale. Note that, with $L < 200$, the analytic result actually gives an upper bound for E[δ]. Given L = 300, λ = 0.1, and $\mu = 0.1$, the analytic and simulation results are the same with $N \leq 60$, as shown in Fig. 3. With $N > 60$, the analytic and simulation results are not the same but they are also very close such that their values are almost equal in logarithmic scale. Note that, with $N > 60$, the analytic result also gives an upper bound for E[δ]. These facts indicate that the probability model is useful for measuring the maximum number of changes of traffic demands δ on the average. Note also that the average value of δ is far less than that of the worst case as derived in Section III.

B. Simulation Results

Simulations are conducted to measure the number of reallocations within the reallocation algorithm, and the interframe dependency.

Fig. 4 shows the simulation results of the average number of reallocations in a call of the reallocation algorithm with various values of N and μ. The results show that it is proportional to N and far less than the number of reallocations in the worst case.

Since we assume that the connection holding time is exponentially distributed with mean $1/\mu$ time slots per connection, the expected number of frames in which a connection lasts is equal to $1/\mu L$. Hence we can use $1/\mu L$ to measure the interframe dependency. Fig. 5 shows the average value of δ as a function of $1/\mu L$ with various values of λ.

The average value of δ decreases as $1/\mu L$ becomes large since most of traffic demands in a given frame are the same as those in the preceding frame. Since the probability values calculated by (7) approach zero except $m = 0$ as $1/\mu L$ becomes large, the average values of δ approach some constants, respectively. This implies that δ will be saturated. This is verified by the simulation results shown in Fig. 5. Meanwhile we can see that the average value of δ is small with interdependent traffic (i.e., $1/\mu L > 1$). That is, the AS algorithm is efficient for interdependent traffic.

V. Conclusions

In previous works, the interdependence between traffic demands in two consecutive frames is neglected for scheduling time slots in a time-division multiplexed switching system. Their approaches to the scheduling problem for avoiding output conflicts are resorted to some combinatorial optimization methods for finding a sequence of matchings. As a result, previous scheduling algorithms have time complexities $O(N^{4.5})$ or $O(N^5)$. However, in this paper, an adaptive scheduling algorithm AS of the worst case time complexity $O(N^2 L)$ is proposed. Comparing the time

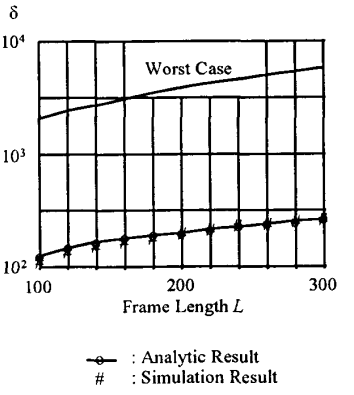

Fig. 2. A comparison of analytic and simulation results with N=20, λ=0.1, and μ=0.1.

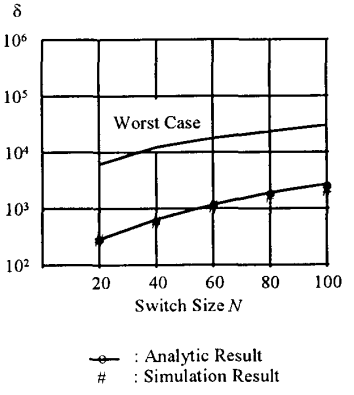

Fig. 3. A comparison of analytic and simulation results with L=300, λ=0.1, and μ=0.1.

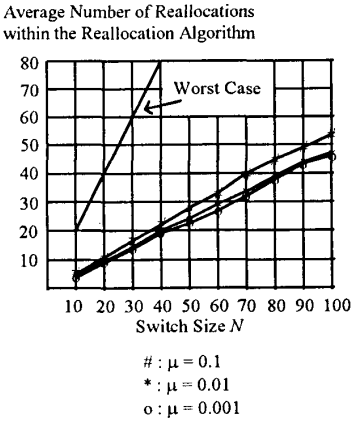

Fig. 4. The average number of reallocations within the reallocation algorithm with L = 50 and λ = 0.5.

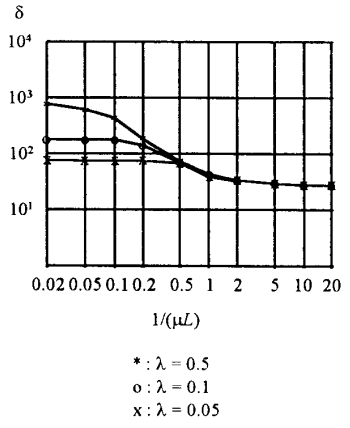

Fig. 5. The average number of changes of traffic demands with N=50 and L=50.

complexity of the AS algorithm with those of previous time slot assignment algorithms, the AS algorithm can perform better than previous time slot assignment algorithms when N is large and/or L is small. More precisely, since the interdependence between traffic demands in two consecutive frames is considered, the performance of the AS algorithm is dependent on the amount of changes of traffic demands. Since the traffic demands in consecutive frames are expected to be interdependent in many practical applications, the AS algorithm may offer as an efficient alternative for scheduling time slots in these applications.

REFERENCES

[1] A. S. Acampora and B. R. Davis, "Efficient Utilization of Satellite Transponders via Time-Division Multibeam Scanning," *Bell Syst. Tech. J.*, Vol. 57, No. 8, pp. 2901-2914, 1978.

[2] G. Bongiovanni, D. Coppersmith, and C. K. Wong, "An Optimum Time Slot Assignment Algorithm for an SS/TDMA System with Variable Number of Transponders," *IEEE Trans. Communications*, Vol. COM-29, pp. 721-726, May 1981.

[3] G. Bongiovanni, D. T. Tang, and C. K. Wong, "A General Multibeam Satellite Switching Algorithm," *IEEE Trans. Communications*, Vol. COM-29, pp. 1025-1036, July 1981.

[4] M. A. Bonuccelli, "A Fast Time Slot Assignment Algorithm for TDM Hierarchical Switching Systems," *IEEE Trans. Communications*, Vol. COM-37, pp. 870-874, Aug. 1989.

[5] W. T. Chen and H. J. Liu, "An Adaptive Scheduling Algorithm for TDM Switching Systems," *Proc. of IEEE INFOCOM'91*, Miami, Florida, USA, April 1991, pp. 668-677.

[6] K. Y. Eng and A. S. Acampora, "Fundamental Conditions Governing TDM Switching Assignments in Terrestrial and Satellite Networks," *IEEE Trans. Communications*, Vol. COM-35, pp. 755-761, July 1987.

[7] T.-Y. Feng, "A Survey of Interconnection Networks," *Computer*, Vol. 14, pp. 12-27, Dec. 1981.

[8] I. S. Gopal, G. Bongiovanni, M. A. Bonuccelli, D. T. Tang, and C. K. Wong, "An Optimal Switching Algorithm for Multibeam Satellite Systems with Variable Bandwidth Beams," *IEEE Trans. Communications*, Vol. COM-30, pp. 2475-2481, Nov. 1982.

[9] I. Gopal, D. Coppersmith, and C. K. Wong, "Minimizing Packet Waiting Time in a Multibeam Satellite System," *IEEE Trans. Communications*, Vol. COM-30, pp. 305-316, Feb. 1982.

[10] I. S. Gopal and C. K. Wong, "Minimizing the Number of Switchings in an SS/TDMA System," *IEEE Trans. Communications*, Vol. COM-33, pp. 497-501, June 1985.

[11] J. E. Hopcroft and R. M. Karp, "An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs," *SIAM J. Comput.*, Vol. 2, pp. 225-231, Dec. 1973.

[12] E. Horowitz and S. Sahni, *Fundamentals of Data Structures in Pascal*. New York: W. H. Freeman and Company, 1990.

[13] T. Inukai, "An Efficient SS/TDMA Time Slot Assignment Algorithm," *IEEE Trans. Communications*, Vol. COM-27, pp. 1449-1455, Oct. 1979.

[14] A. E. Joel, Jr., "Digital Switching-How it has developed," *IEEE Trans. Communications*, Vol. COM-27, pp. 948-959, July 1979.

[15] L. Kleinrock, *Queueing Systems, Vol. 1: Theory*. New York:Wiley, 1975.

[16] E. L. Lawler, *Combinatorial Optimization: Networks and Matroids*. New York: Holt, Rinehart and Winston, 1976.

[17] J. L. Lewandowski, J. W. S. Liu, and C. L. Liu, "SS/TDMA Time Slot Assignment with Restricted Switching Modes," *IEEE Trans. Communications*, Vol. COM-31, pp. 149-154, Jan. 1983.

[18] C. L. Liu, *Introduction to Combinatorial Mathematics*. New York: McGraw-Hill, 1968.

[19] C. A. Pomalaza-Raez, "A Note on Efficient SS/TDMA Assignment Algorithms," *IEEE Trans. Communications*, Vol. COM-36, pp. 1078-1082, Sept. 1988.

[20] R. Ramaswamy and P. Dhar, "Comments on "An Efficient SS/TDMA Time Slot Assignment Algorithm"," *IEEE Trans. Communications*, Vol. COM-32, pp. 1061-1065, Sept. 1984.

[21] C. Rose, "Rapid Optimal Scheduling for Time-Multiplex Switches Using a Cellular Automaton," *IEEE Trans. Communications*, Vol. COM-37, pp. 500-509, May 1989.

[22] C. Rose and M. G. Hluchyj, "The performance of Random and Optimal Scheduling in a Time Multiplex Switch," *IEEE Trans. Communications*, Vol. COM-35, pp. 813-817, Aug. 1987.

[23] T. Scarcella and R. V. Abbott, "Orbital Efficiency Through Satellite Digital Switching," *Communications*, pp. 38-46, May 1983.

Wen-Tsuen Chen(M'87-SM'89-F'94) was born in Taiwan on May 27, 1948. He received the B.S. degree in nuclear engineering from National Tsing Hua University, Taiwan, Republic of China, and the M.S. and Ph.D. degrees in electrical engineering and computer sciences both from University of California at Berkeley, in 1970, 1973, and 1976, respectively.

He joined the faculty of the Institute of Applied Mathematics, National Tsing Hua University in March 1976 as an Associate Professor. Since August 1979, he has been a Professor of the Department of Computer Science, National Tsing Hua University. From 1983 to 1988 he served as the Chairman of the Department. In 1980, he was a Visiting Professor in the Department of Electrical Engineering and Computer Sciences of the University of California at Berkeley. Since 1988, he has been a member of the Science and Technology Advisory Office of the Ministry of Education, Republic of China and is currently on leave from the National Tsing Hua University to serve as the Director of the above Advisory Office. His current research interests include computer networks, broadband ISDN, multiprocessing systems, and parallel algorithms.

Dr. Chen was a recipient of the Distinguished Computer Professionals Award from the Ministry of the Economic Affairs in 1984, received the Outstanding Achievement Award from the Ministry of Defense in 1987, and the Technology Innovation Award from the Sun Yat-Sen Academics and Culture Foundation in 1989, and was a recipient of the Distinguished Research Award from the National Science Council in 1990 and 1992. From 1984 to 1985, he was elected as an IEEE Distinguished Visitor in region 10. From 1991 to 1992, he served as a member of the Board of Directors of the IEEE Taipei Section.

Dr. Chen is also a member of the Association for Computing Machinery, the Chinese Institute of Engineers, the Chinese Institute of Electrical Engineeing, and Phi Tau Phi.

Huai-Jen Liu received the B.E. degree with the first place in information and computer engineering from Chung Yuan Christian University, Taiwan, Republic of China, the M.S. and Ph.D. degrees in computer science both from National Tsing Hua University, Taiwan, Republic of China, in 1986, 1988, and 1992, respectively. From 1992 to 1994, he was in military service with the responsibility of network planning, construction, and maintenance. He is currently with the Telecommunication Laboratories, Taiwan, Republic of China, as an Associate Research Engineer. His current research interests include ATM switching networks and traffic control for B-ISDN.

Dr. Liu is a member of Phi Tau Phi.